Application du Decorator

*A travers les exercices de ce TD, vous allez décorer les classes dérivées de la classe **Engine** et de la classe **Hull** en vous aidant du Design Pattern Decorator.*

Exercice 1 :

Cette partie introduit la mise en pratique du Pattern Decorator à la classe Engine afin de modifier l'implémentation de cette classe sans toucher à son code.

1°) Ecrire une classe abstraite **EngineComponent** qui hérite de la classe **Engine.** Cette classe comporte un pointeur vers un objet de type **Engine** qui correspondra à l'objet décoré. La classe doit redéclarer également la méthode *getSpeed()* en virtuelle pure.

2°) Faire hériter la classe **EngineComponent** par 2 autres classes de votre choix. Ré-implémenter la méthode *getSpeed()* dans chacune des sous-classes sachant que :

- La première classe double la valeur de la vitesse d'origine.
- La seconde classe augmente la valeur d'origine de 50% de la valeur de son carré.

Exercice 2 :

Cette partie introduit la décoration de la classe Hull afin de modifier l'implémentation de cette classe sans toucher à son code.

1°) Ecrire une classe **HullComponent** qui hérite de la classe **Hull**. Cette classe comporte un pointeur vers un objet de type **Hull** qui correspondra à l'objet décoré. La classe doit redéclarer également la méthode *getSolidity()* en virtuelle pure.

2°) Faire hériter la classe **HullComponent** par une autre classe. Ré-implémenter la méthode *getSolidity()* en prenant exemple sur l'exercice précédent.

Exercice 3 :

1°) A l'aide du Design Pattern Décorateur, modifier la Factory permettant de créer des objets de type **Hull** et de type **Engine** en enveloppant les objets créés avec les objets qui héritent des classes abstraites **HullComponent** et **EngineComponent** définies précédemment.

2°) Dans cette même Factory, inventer de nouvelles combinaisons d'objets à l'aide de vos décorateurs.